REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
10-17-2005

2. REPORT TYPE
Final

3. DATES COVERED *(From - To)*
23-9-2002 to 14-9-2005

4. TITLE AND SUBTITLE
Magnetoelectronic Reconfigurable Logic

5a. CONTRACT NUMBER

5b. GRANT NUMBER
N00014-02-1-1038

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Nicholas P. Carter

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
The Board of Trustees of the University of Illinois at Urbana-Champaign
1901 S. First St., Suite A
Champaign, IL 61820-7406

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Office of Naval Research
One Liberty Center
875 North Randolph Street, Suite 1425
Arlington, VA 22203-1995

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Circuit elements that combine ferromagnetic materials with semiconductor structures have the potential to overcome two of the most significant limitations of CMOS systems: data loss on power failure and radiation-induced soft errors. Unlike CMOS structures, which rely on capacitatively-stored charge to store data, these magnetoelectronic devices encode binary values using the magnetization directions of their ferromagnetic elements, which retain their state without power. In this project, we have developed a number of circuit and system architectures that exploit the properties of a particular magnetoelectronic device, the hybrid Hall effect device, to deliver non-volatile operation and high performance. At the circuit level, we have developed designs for reconfigurable logic gates based on the HHE device that perform both AND/OR and threshold computations. Our system designs integrate non-volatile magnetic memories into processor architectures to produce self-checkpointing microprocessors that recover near-instantly from power failures and outperform conventional architectures in many cases.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Nicholas P. Carter
					19b. TELEPHONE NUMBER *(Include area code)*
U	U	U	UU	10	(217) 244-7162

INSTRUCTIONS FOR COMPLETING SF 298

1. **REPORT DATE.** Full publication date, including day, month, if available. Must cite at least the year and be Year 2000 compliant, e.g. 30-06-1998; xx-06-1998; xx-xx-1998.

2. **REPORT TYPE.** State the type of report, such as final, technical, interim, memorandum, master's thesis, progress, quarterly, research, special, group study, etc.

3. **DATES COVERED.** Indicate the time during which the work was performed and the report was written, e.g., Jun 1997 - Jun 1998; 1-10 Jun 1996; May - Nov 1998; Nov 1998.

4. **TITLE.** Enter title and subtitle with volume number and part number, if applicable. On classified documents, enter the title classification in parentheses.

5a. **CONTRACT NUMBER.** Enter all contract numbers as they appear in the report, e.g. F33615-86-C-5169.

5b. **GRANT NUMBER.** Enter all grant numbers as they appear in the report, e.g. AFOSR-82-1234.

5c. **PROGRAM ELEMENT NUMBER.** Enter all program element numbers as they appear in the report, e.g. 61101A.

5d. **PROJECT NUMBER.** Enter all project numbers as they appear in the report, e.g. 1F665702D1257; ILIR.

5e. **TASK NUMBER.** Enter all task numbers as they appear in the report, e.g. 05; RF0330201; T4112.

5f. **WORK UNIT NUMBER.** Enter all work unit numbers as they appear in the report, e.g. 001; AFAPL30480105.

6. **AUTHOR(S).** Enter name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. The form of entry is the last name, first name, middle initial, and additional qualifiers separated by commas, e.g. Smith, Richard, J, Jr.

7. **PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES).** Self-explanatory.

8. **PERFORMING ORGANIZATION REPORT NUMBER.** Enter all unique alphanumeric report numbers assigned by the performing organization, e.g. BRL-1234; AFWL-TR-85-4017-Vol-21-PT-2.

9. **SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES).** Enter the name and address of the organization(s) financially responsible for and monitoring the work.

10. **SPONSOR/MONITOR'S ACRONYM(S).** Enter, if available, e.g. BRL, ARDEC, NADC.

11. **SPONSOR/MONITOR'S REPORT NUMBER(S).** Enter report number as assigned by the sponsoring/ monitoring agency, if available, e.g. BRL-TR-829; -215.

12. **DISTRIBUTION/AVAILABILITY STATEMENT.** Use agency-mandated availability statements to indicate the public availability or distribution limitations of the report. If additional limitations/ restrictions or special markings are indicated, follow agency authorization procedures, e.g. RD/FRD, PROPIN, ITAR, etc. Include copyright information.

13. **SUPPLEMENTARY NOTES.** Enter information not included elsewhere such as: prepared in cooperation with; translation of; report supersedes; old edition number, etc.

14. **ABSTRACT.** A brief (approximately 200 words) factual summary of the most significant information.

15. **SUBJECT TERMS.** Key words or phrases identifying major concepts in the report.

16. **SECURITY CLASSIFICATION.** Enter security classification in accordance with security classification regulations, e.g. U, C, S, etc. If this form contains classified information, stamp classification level on the top and bottom of this page.

17. **LIMITATION OF ABSTRACT.** This block must be completed to assign a distribution limitation to the abstract. Enter UU (Unclassified Unlimited) or SAR (Same as Report). An entry in this block is necessary if the abstract is to be limited.

Final Report for Award N00014-02-1-1038, "Magnetoelectronic Reconfigurable Logic"

Nicholas P. Carter
Coordinated Science Laboratory, University of Illinois at Urbana-Champaign
1308 West Main St.
Urbana, IL 61801
phone: (217) 244-7162 fax: (217) 244-1764
Email: npcarter@crhc.uiuc.edu

Abstract

Circuit elements that combine ferromagnetic materials with semiconductor structures have the potential to overcome two of the most significant limitations of CMOS systems: data loss on power failure and radiation-induced soft errors. Unlike CMOS structures, which rely on capacitatively-stored charge to store data, these magnetoelectronic devices encode binary values using the magnetization directions of their ferromagnetic elements, which retain their state without power. In this project, we have developed a number of circuit and system architectures that exploit the properties of a particular magnetoelectronic device, the hybrid Hall effect device, to deliver non-volatile operation and high performance. At the circuit level, we have developed designs for reconfigurable logic gates based on the HHE device that perform both AND/OR and threshold computations. Our system designs integrate non-volatile magnetic memories into processor architectures to produce self-checkpointing microprocessors that recover near-instantly from power failures and outperform conventional architectures in many cases.

I. Introduction

Data volatility is a significant problem in many electronic systems. The memory elements commonly used in CMOS devices, including SRAM cells, DRAM cells, and register latches, rely on capacitatively-stored charge to represent binary data. When power is removed from the chip, this charge quickly drains off, destroying any data stored in the memory. This leads to a number of problems, including loss of data when power failures occur, long power-on/boot times because operating systems must be loaded from disk each time the system is turned on, and high standby power consumption due to leakage currents.

Magnetoelectronic devices, which combine ferromagnetic materials with conventional semiconductor structures, have the potential to overcome this limitation of CMOS electronics by providing designers with high-performance, non-volatile memory and logic gates that can be tightly integrated with CMOS circuitry. These devices use the magnetization direction of one or more ferromagnetic elements to store data, making them both inherently non-volatile and highly resistant to radiation-induced soft errors. Unlike competing non-volatile memory technologies, magnetoelectronic devices both operate at sub-nanosecond speeds and support arbitrary numbers of read/write cycles without wearing out, making them good candidates for computing applications.

This report summarizes the results of work done under grant N00014-02-1-1038, "Magnetoelectronic Reconfigurable Logic," which explored system and circuit-level applications for the hybrid Hall effect (HHE) device. This effort initially focused on circuit-level designs of reconfigurable logic gates. Later phases of the work focused on system-level applications, and developed architectures for self-checkpointing processors that tolerate power failures by periodically copying critical program state to on-chip non-volatile memories.

The body of this report follows the chronological progress of our work. We first present a brief overview of the HHE device and its operation. Next, we describe our HHE-based reconfigurable logic gates. This is followed by a discussion of our self-checkpointing microprocessor architectures, and a conclusion.

II. The Hybrid Hall Effect Device

The designs presented in this paper are based on the hybrid Hall effect (HHE) device, a magnetoelectronic circuit element developed at the Naval Research Lab. Figure 1(a) shows an atomic force micrograph of a HHE device that was fabricated at the Naval Research Laboratory in Washington D.C, while 1(b) illustrates its operation. Current

TABLE I

HHE DEVICE CHARACTERISTICS

Parameter	Value	Comments
Area	$35f^2$	Area of an SRAM cell is approximately 150 f^2 in a process with feature size f
Write current	10 mA	Scales approximately linearly with the feature size.
Read current	10 mA	
Write/Read times	500 ps	100 ps read/write times expected with future HHE Devices.

flowing through the write wire at the top of the figure (not shown in the AFM), creates a magnetic field. If the intensity of the magnetic field exceeds the magnetization threshold of the ferromagnetic bar in the middle of the device, it magnetizes the device in either the left or right directions, depending on the direction of current flow.

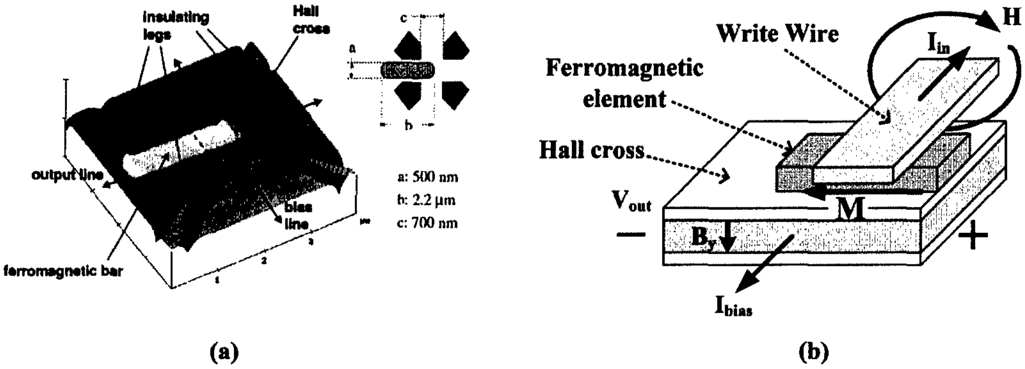

Fig. 1. Hybrid Hall effect device. (a) Atomic force micrograph (figure courtesy Mark Johnson, Naval Research Laboratory) (b) Operation

To read the state of the HHE device, a bias current I_{bias} is passed through the bias line of the conductor under the ferromagnetic bar, such that the current passes under the edge of the ferromagnetic bar, where the magnetic field generated by the bar is nearly vertical. When the bias current encounters this magnetic field, the Hall effect induces a Hall output voltage that is perpendicular to both the bias current and the magnetic field. For a given direction of bias current flow, the sign of this output voltage is determined by the magnetization direction of the ferromagnetic bar, and the magnitude of the output voltage is determined by the product of the magnitude of the bias current and the Hall resistance of the HHE device, which, for current devices, is approximately 10 Ohms. Table I lists the parameters of the HHE device that are most relevant to system designers such as its physical size, the amount of current required to set the state of the device (write current), and the device's speed.

The HHE device has three advantages over competing technologies: speed, long lifetimes, and ease of fabrication. HHE devices have been shown to operate in as little as 500ps, which compares extremely well to the microsecond or millisecond write times of FLASH memory cells. Unlike Ovonic or FeRAM devices, HHE devices can be written an arbitrary number of times without damaging the device, which is particularly important in computing applications. Finally, HHE devices require only a single layer of ferromagnetic material, making fabrication processes for HHE-based systems less costly than fabrication processes for most other magnetoelectronic devices, which require multiple layers of ferromagnetic material.

III. MAGNETOELECTRONIC LOGIC GATES

The first 18 months of this contract developed designs for reconfigurable logic gates based on the HHE device. These designs treat the HHE device as a computing element that evaluates the current flowing through its input wire and changes state if that current exceeds a threshold value, rather than as a memory device that stores a bit of data. They consume power while evaluating their inputs and while their outputs are being read, but hold their results indefinitely without consuming power, even across interruptions to the system's power supply.

Figure 2 presents a schematic for an HHE-based reconfigurable logic gate developed during this project. In this circuit, the X_{1-4} inputs are the inputs to the logic gate, and drive transistors sized to flow approximately 25% of the current required to change the state of the logic gate. The C_1 input is a configuration bit, and drives a transistor sized such that it flows 75% of the current required to change the state of the device. Thus, if C_1 is asserted, only one of the X_{1-4} inputs needs to be asserted to set the output of the gate to "1", and the gate computes the OR of its inputs. If C_1 is not asserted, all of the X_{1-4} inputs need to be asserted to set the gate to "1," and it computes the AND of its inputs. If this gate is extended to conditionally invert the X_{1-4} inputs based on a second configuration input C_2, it can be reconfigured on a cycle-by-cycle basis to compute the AND, OR, NAND, or NOR of its inputs.

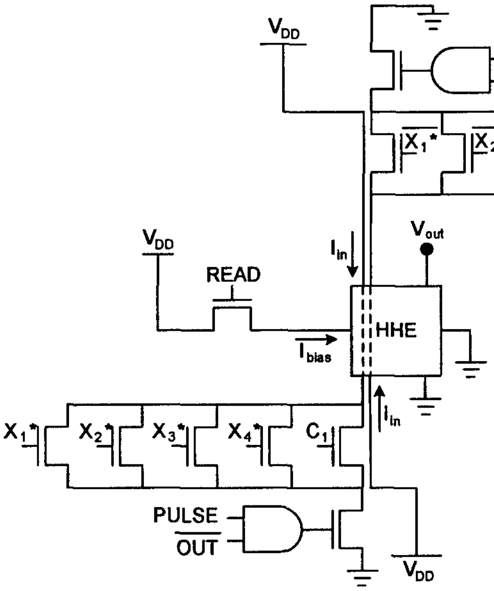

Fig. 2. HHE-based reconfigurable logic gate

Power consumption is a major issue in HHE-based logic gates, as they require large (10mA) currents to perform read and write operations. Our designs address this by limiting the durations of the read (bias) and write currents as much as possible while ensuring correct operation. The logic gate shown in Figure 2 uses output feedback to limit the duration of write currents. This gate assumes that the HHE device has two write wires, a common practice in magnetoelectronic devices. Current flows through the left-hand write wire in the direction that sets the output of the gate to "1," and through the right-hand wire in the direction that sets the output to "0." Current flow through each write wire is gated by the AND of a PULSE signal that is active for 500ps following each transition on the inputs and either the true or inverted output of the gate. This causes current to flow through the write wires only when the gate's inputs are being evaluated, and prevents any current flow through a write wire if the gate's output is already in the state that that wire could set it to. When the gate's output is "1", the left-hand write wire is gated off, as it is not necessary to re-evaluate the gate's inputs in order to maintain the output state. Similarly, the right-hand write wire is gated off when the output of the gate is "0" to prevent unnecessary current flow through that wire.

Figure 3 shows an HSPICE simulation of the gate in Figure 2, showing that it can be dynamically reconfigured to support AND, OR, NAND, and NOR operations. In this figure, the READ (bias) input is asserted 10ns after each input change, so the gate's output lags 10ns behind the inputs. For example, the output change at t=50ns is triggered by the input change at t=40ns, and so on. Given the speed of the HHE device, this input-read delay could have been as low as 0.5ns, but we used a 10ns spacing for simplicity. We have also demonstrated that this gate retains its output state across power supply interruptions, resuming its original output when power is restored.

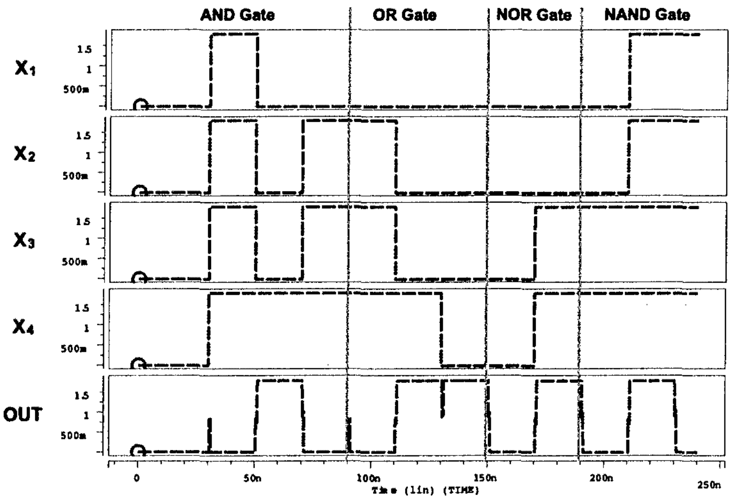

Fig. 3. Operation of the HHE-based reconfigurable logic gate

To further reduce power consumption, we have developed the output interface shown in Figure 4. This circuit uses an SRAM cell to both amplify and latch the output of the HHE device, reducing both the amount of bias current that must be applied and the duration of the bias current. To read the output of the HHE device, the SENSE input is asserted, forcing the HHE device into its metastable state, and the bias current I_{bias} applied. Once the output of the HHE device has stabilized, SENSE is de-asserted, and the SRAM cell converges to one of its two stable states, depending on the output voltage of the HHE device (V_{out}), at which time I_{bias} is removed. Our simulations show that this interface correctly senses HHE device output voltages of 100mV, allowing the use of 10mA bias currents as opposed to the 100mV bias currents that would be required to generate 1V CMOS-compatible output voltages and eliminating the need to drive the bias current during times when the gate's output is stable.

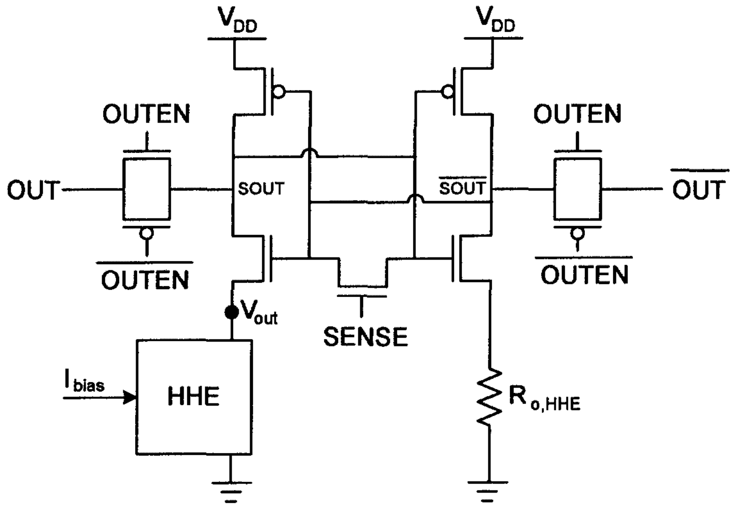

Fig. 4. CMOS-compatible output interface

In addition to AND/OR/NAND/NOR gates, we developed the threshold logic gates shown in Figure 5. In threshold logic, each gate's output depends on whether the number of its inputs that are "1" exceeds a specific value, called the gate's *threshold*. Threshold logic is a superset of AND/OR logic, in that OR can be implemented as a threshold gate with a threshold of 1, and an n-input AND can be implemented as a threshold gate with a threshold of n.

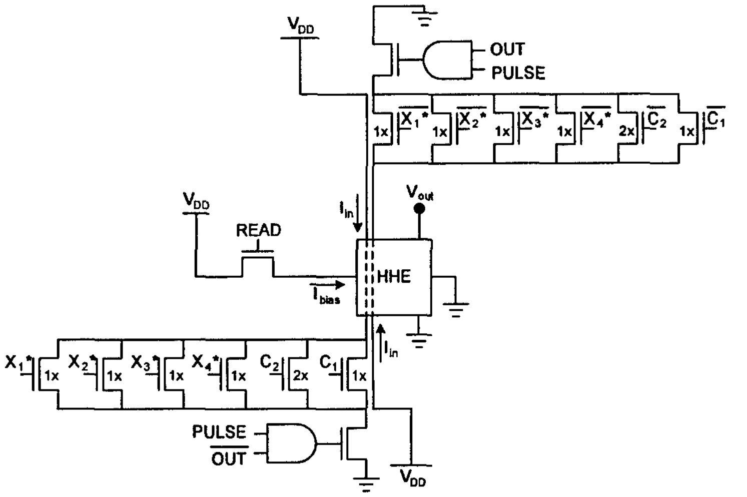

Fig. 5. HHE-based threshold logic gate

Our gate designs exploit the properties of the HHE device to implement threshold logic much more efficiently than is possible in CMOS systems. HHE devices are inherently threshold-based, in that their output changes if the magnitude of their write current exceeds a fixed value. Thus, we can convert an n-input AND/OR/NAND/NOR HHE gate into a threshold gate by adding $\log_2(n) - 1$ configuration inputs to the gate and appropriately sizing the transistors driven by those inputs. In Figure 5, which shows a four-input threshold gate, the C_1 and X_{1-4} inputs drive transistors sized to flow 25% of the current required to set the state of the gate, while the C_2 input drives a transistor that is twice as large. As a result, the gate's output is "1" if $X_1 + X_2 + X_3 + X_4 + (2 * C_2) + C_1 \geq 4$.

The main advantage of threshold logic over AND/OR logic is that fewer threshold gates are required to implement many functions than AND/OR gates. Figure 6 shows the number of 4-input threshold and AND/OR gates required to implement all possible 4-input Boolean functions. On average, threshold logic requires 4 gates to implement a function, while AND/OR logic requires 5, justifying the increase in the number of transistors required to implement threshold gates as opposed to AND/OR gates.

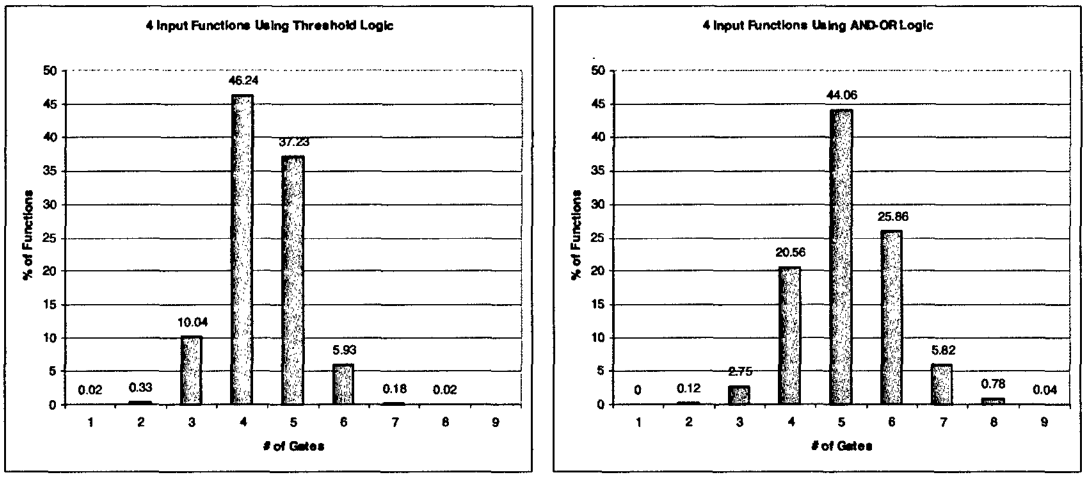

Fig. 6. Number of gates required to implement 4-input functions using threshold or AND/OR logic

Our gate-level designs have demonstrated that HHE devices can implement non-volatile logic gates that can operate on CMOS signal levels and generate CMOS-compatible outputs. As part of this effort, we designed input interfaces that convert CMOS voltages into the currents required by HHE devices, and output interfaces that amplify and latch the voltage generated by the HHE device. These interfaces limit the magnitude and duration of the HHE device's write and bias currents to reduce power consumption while still providing non-volatile operation. In addition, we have shown that the HHE device can efficiently implement threshold logic, allowing many Boolean functions to be implemented in fewer gates than is possible in AND/OR designs.

IV. Architecture of a Self-Checkpointing Microprocessor

The second half of this effort focused on system-level applications of magnetoelectronic devices, in particular their use in developing microprocessors that can tolerate power failures without losing data and can provide "instant-on" operation, eliminating the long "boot" times seen in current computer systems. The HHE device's high power consumption makes it impractical to construct integrated circuits completely out of HHE-based devices. Instead, our designs add a relatively-small number of magnetoelectronic devices to a CMOS processor that store the critical state of an application running on the processor. If the processor's power supply is interrupted, the application's state can be restored from the magnetoelectronic devices, allowing it to resume execution with little or no loss of progress.

Figure 7 shows a block diagram of the self-checkpointing microprocessor that we have developed. This design adds four magnetoelectronic memories to a conventional microprocessor: non-volatile copies of the program counter and register file, a checkpoint buffer that holds all data written to the memory since the last checkpoint, and a dirty data buffer that holds a copy of all data that is dirty in the processor's cache (i.e., has been written since it was brought into the cache).

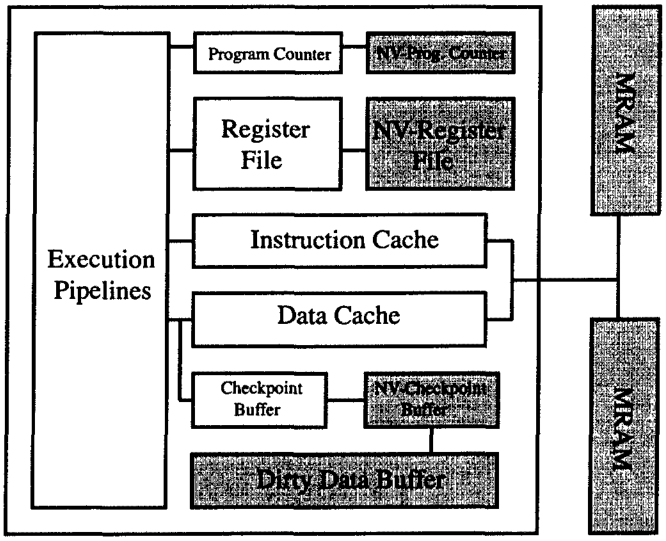

Fig. 7. A self-checkpointing processor.

During normal execution, the self-checkpointing processor fetches and executes instructions in the same way a conventional microprocessor does. Periodically, it *checkpoints* its state by copying the contents of the register file and program counter into the non-volatile versions of those structures and saving a copy of all data that has been written to memory since the last checkpoint in the non-volatile portion of the checkpoint buffer. Between checkpoints, the processor moves the contents of the non-volatile portion of the checkpoint buffer into the dirty data buffer to free up space for the next checkpoint.

In combination with non-volatile off-chip memory implemented using MRAMs or a similar technology, this checkpointing scheme allows the processor to tolerate power failures by "rolling back" execution to the last checkpoint when the power supply is restored after an interruption. When power is restored, the processor copies the contents of the checkpoint buffer and dirty data buffer into the off-chip memory, making the off-chip memory's contents match the state of the memory system when the checkpoint was taken. The processor then loads the program counter and register file from their non-volatile copies, and resumes execution of the program.

Self-checkpointing processors have a number of advantages over conventional microprocessors. They protect applications against data loss during power supply interruptions, which can be very important for long-running computations. Perhaps more importantly for military applications, they can provide "instant-on" operation, by avoiding the need to "boot" the operating system when a device is powered on. Since a self-checkpointing processor retains its state when powered down, it can simply resume execution of its operating system and programs when powered on, rather than having to load an OS from disk, configure any peripherals attached to the system, and initialize memory before program execution can begin. As leakage currents become a dominant contributor to integrated circuit power consumption in future fabrication processes, this "instant-on" capability will also reduce overall power consumption by allowing systems to power completely off when not in use and resume operation immediately when needed.

A. Performance Analysis

To evaluate our mechanisms for self checkpointing, we implemented a self-checkpointing version of the Pentium 4 [1] microprocessor in simulation. Figure 8 shows the performance of the self-checkpointing processor relative to the baseline architecture on a suite of programs taken from the SPEC and MediaBench suites, as a function of the processor's second-level (L2) cache size and the capacity of the dirty data buffer. In this graph, a performance value of "1" indicates that the self-checkpointing architecture ran the program in the same number of cycles as the base architecture, higher values indicate that the self-checkpointing processor executed the program in less time than the base, and values less than one indicate that the self-checkpointing processor took more time to execute the program than the original design.

In many cases, particularly when the processor had a large L2 cache, the self-checkpointing processor was slightly slower than the baseline architecture, as would be expected. While our design takes only 1-2ns to perform a checkpoint, checkpointing does add some overhead to the execution time. The worst-case performance of the self-checkpointing processor was 0.7% slower than the base design, showing that the overheads of our checkpointing hardware are very small.

Interestingly, however, the self-checkpointing processor outperforms the baseline design on a large number of programs. This performance advantage is most pronounced when the system has a small L2 cache, and when the dirty data buffer is relatively small (2-4 KB). This result was unexpected, and is due to the way in which the self-checkpointing processor manages its cache. In order to restore a program's state after a power failure, the checkpoint and dirty data buffers must contain all of the data that has been modified since it was brought into the processor's cache. If the amount of modified (dirty) data in the processor's cache exceeds the capacity of the dirty data buffer, the self-checkpointing processor copies some of the dirty data back to the off-chip memory, leaving the line containing the data in the cache but marking it clean.

Writing lines back to the off-chip memory in this way improves performance by eliminating the need to write them back when the processor needs to replace the line in the cache with some other line from memory. Cache misses in computer programs tend to occur in bursts. During those bursts, there is a lot of demand for the off-chip memory, and the performance of the program is limited by how much data must be moved into or out of the cache. By writing some of the cache's dirty data back to the off-chip memory during times when the memory system is idle, the self-checkpointing processor reduces the amount of data that must be written to the off-chip memory during bursts of cache misses, increasing overall performance. This "early writeback" effect has been noticed by other computer architects, a number of whom have attempted to design memory systems to take advantage of it, and is a natural side-effect of our architecture.

B. Power Consumption

To estimate the power consumed by our magnetoelectronic memories, we counted the number of writes to each memory during the execution of a program and multiplied by the estimated energy cost of each write. This gave the

[1] Pentium 4 is a trademark of the Intel Corporation

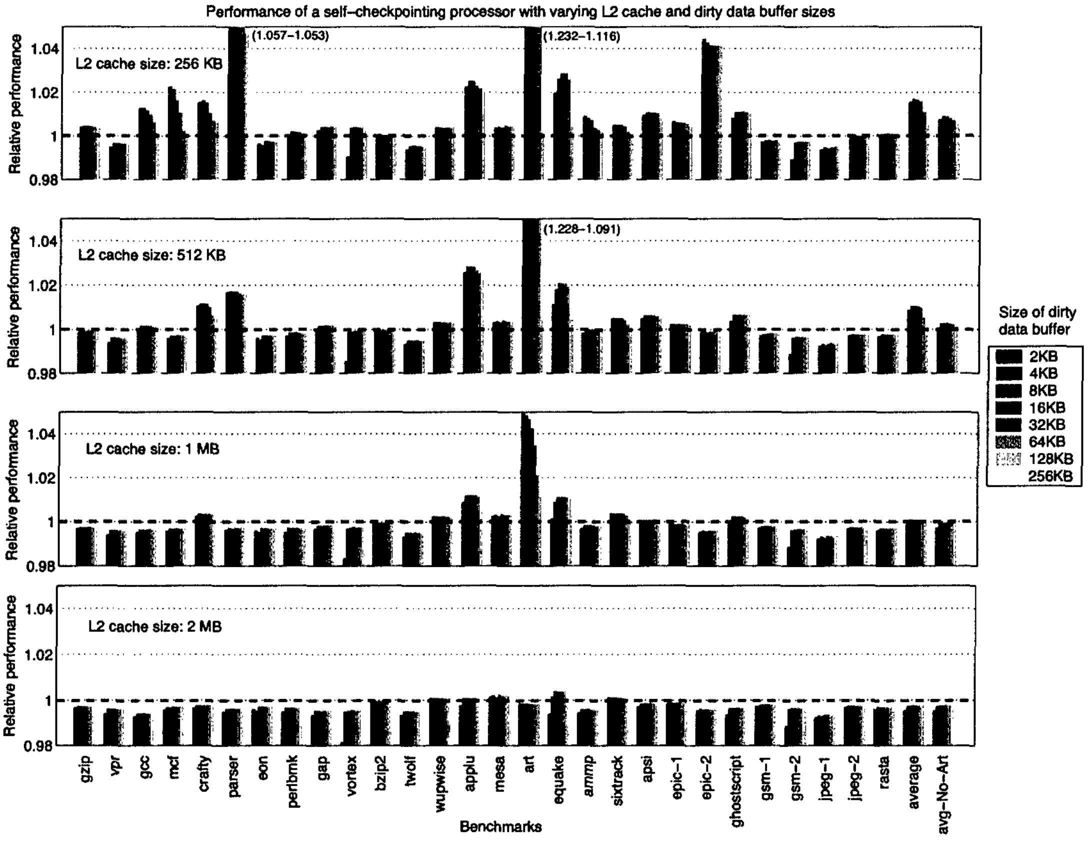

Fig. 8. Performance of a self-checkpointing processor for different L2 cache sizes with a 512-byte eight-way associative checkpoint buffer and finite-sized dirty data buffer.

total energy consumed by the magnetoelectronic memories, which we divided by the program's simulated execution time to get the average power consumption. We calculated power consumption using three different estimates of the energy cost per write to a magnetoelectronic device: worst-case, conservative, and optimistic. For the worst-case estimate, we assumed that the current required to read or write each HHE device remained 10 mA, as observed in prototype devices, and that each bit's write current passed directly from the supply rail to ground through that bit (i.e., no chaining of write currents to reduce power). Similarly, we assumed that the read and write currents must be applied for 500 picoseconds to sense or set the state of the device.

In our conservative estimates, we assumed that blocks of eight memory cells were chained together to share read and write currents, reducing the power consumption by a factor of eight. Commercial MRAMs share read and write currents across as many as 128 bit cells, so chaining groups of eight bit cells in a microprocessor environment is unlikely to be a significant problem. Our optimistic estimates also assume chaining of eight-cell blocks, and assume that scaling HHE devices from feature sizes of 0.5 microns to 0.1 microns to match the feature sizes of current-generation microprocessors also reduces read and write currents from 10mA to 2mA by scaling the width of the write wire in each device and improving the read sensitivity of the device.

Figure 9 shows the average power consumed in the magnetoelectronic memories when our benchmarks are run on an architecture with a 512 KB L2 cache, a 512 B checkpoint buffer, and an 8 KB dirty data buffer. Under our conservative assumptions, the average power consumed during benchmark execution is only 62mW, a small fraction of the power consumed by the base architecture. Under our optimistic assumptions, power consumption drops further, to 12.3 mW. Even using our worst-case estimates, the power consumed by our non-volatile memories is under 500mW.

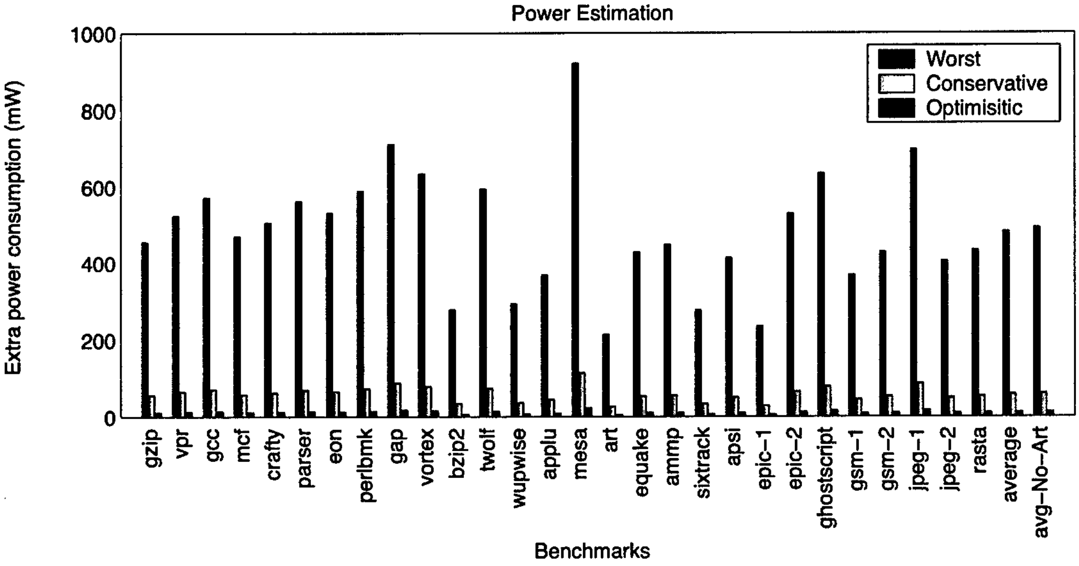

Fig. 9. Power costs of self-checkpointing structures.

C. Area Costs

The primary area cost of our mechanisms for self-checkpointing is the area required by the magnetoelectronic and conventional memories required to implement our mechanisms. On the Pentium 4, the non-volatile register file requires 1806 bytes of non-volatile memory to hold the register set described in the last section. A 512-byte checkpoint buffer contains 128 entries of 32 bits each and requires approximately 1 KB of non-volatile and conventional memory for data and address (tag) storage.

Similarly, an 8KB dirty data buffer has 2K entries and requires 16 KB of non-volatile storage, as well as a 32-bit addressed, N entry pipelined NAND-CAM, where N is the number of data entries in the dirty data buffer. Each entry in the NAND-CAM requires $33 + \log_2(N)$ bits of memory to hold its valid bit, tag, and location pointer, for a total of 11.5KB of storage.

Thus, the total area cost of our structures is approximately 19 KB of non-volatile memory and 15 KB of SRAM. Using the circuits we have developed, each bit of magnetoelectronic storage will take up approximately twice as much area as an SRAM bit, making the total area cost equivalent to approximately 53KB of SRAM memory, or about 10% of the area required by the Pentium 4's L1 and L2 caches.

V. Publications Supported by This Grant

A. Conference and Journal Papers

- Love Kothari and Nicholas P. Carter. "Using Nanomagnetic Devices to Tolerate Data Volatility in Microprocessors" *Under review, IEEE Transactions on Computers*
- Nicholas P. Carter, Steven Ferrera, Love Kothari, and Stanley Ye. "Hall-Effect Circuits and Architectures for Non-Volatile System Design" *European Conference on Circuit Theory and Design, September 2005*
- Steve P. Ferrera and Nicholas P. Carter. "Reconfigurable Magnetoelectronic Circuits for Threshold Logic" *International Journal of Circuit Theory and Applications, November 2004*
- Steve P. Ferrera and Nicholas P. Carter. "A Magnetoelectronic Macrocell Employing Reconfigurable Threshold Logic" *Proceedings of the 12th ACM International Symposium on Field-Programmable Gate Arrays (FPGA). February 2004*
- Steve P. Ferrera, Nicholas P. Carter. "Reconfigurable Circuits Using Hybrid Hall Effect Devices" *Proceedings of the 13th International Conference on Field Programmable Logic and Applications (FPL). September 2003.*

- S. Ferrera and N.P. Carter. "Hybrid Hall Effect Devices – A Novel Building Block for Reconfigurable Logic" *Second Workshop on Non-Silicon Computation, June 2003*

B. Theses

- "The Design of a Nonvolatile Processor using Magnetoelectronic Devices." Love Kothari. Master's Thesis, University of Illinois at Urbana-Champaign, April 2005.
- "Reconfigurable Circuit Designs and Applications for Magnetoelectronic Logic." Steve P. Ferrera. Master's Thesis, University of Illinois at Urbana-Champaign, May 2004.

VI. Conclusion

Magnetoelectronic devices have the potential to enable the design of electronic systems with capabilities that are unachievable in conventional CMOS – high-performance, non-volatile storage, the ability to tolerate power failures without losing data, instant-on operation, and zero idle power consumption. To date, most of the work in magnetoelectronics has been done at the device level, developing and fabricating devices that combine ferromagnetic materials and semiconductor structures. System-level work has tended to focus on memory applications, such as MRAMs, and there has been little investigation of designs that use magnetoelectronics to implement logic or that integrate magnetoelectronic storage with CMOS circuitry.

This project has explored the integration of magnetoelectronics with logic at both the circuit and system levels. Our early work developed reconfigurable magnetoelectronic logic gates, combining HHE devices with CMOS transistors to create gates that retain their output values indefinitely, even in the absence of an external power supply. To limit power consumption, which is a significant issue in magnetoelectronic systems, these designs use output feedback and timing pulses to prevent current from flowing through the HHE device except when absolutely necessary. We simulated both AND/OR and threshold logic gates, showing that HHE devices could be used to implement threshold logic efficiently.

At the system level, we developed an architecture for a self-checkpointing processor that tolerates power failures by periodically copying critical program state to on-chip non-volatile memories. Our self-checkpointing structures increase the amount of area required for on-chip memory in a Pentium 4 processor by approximately 10%, and increase power consumption by only 62mW. In our experiments, a self-checkpointing version of the Pentium 4 performed at most 0.7% worse than the baseline architecture, and outperformed the base processor in most cases, because the self-checkpointing hardware forces the system to use on-chip memory more efficiently.

In our future work, we plan to continue to close the gap between device-level and system-level magnetoelectronics. Our immediate plans focus on designing and demonstrating medium-scale magnetoelectronic structures, such as logic arrays or portions of our mechanisms for self-checkpointing. Within a 5-6 year timeframe, we expect to be able to fabricate a prototype self-checkpointing processor, demonstrating that magnetoelectronic devices can be integrated with conventional electronics at the system level.